(54) Title: USING UNPREDICTABLE INFORMATION TO MINIMIZE LEAKAGE FROM SMARTCARDS AND OTHER CRYPTOSYSTEMS

(57) Abstract

Methods and apparatuses are disclosed for securing cryptosystems against external monitoring attacks by reducing the amount (and signal to noise ratio) of useful information leaked during processing. This is generally accomplished by incorporating unpredictable information into the cryptographic processing. Various embodiments of the invention use techniques such as reduction of signal to noise ratios, random noise generation (101, 105), clock skipping (240), and introducing entropy into the order of processing operations or the execution path. The techniques may be implemented in hardware or software, may use a combination of digital and analog techniques, and may be deployed in a variety of cryptographic devices.

5   # USING UNPREDICTABLE INFORMATION TO MINIMIZE LEAKAGE FROM SMARTCARDS AND OTHER CRYPTOSYSTEMS

This application claims the benefit of US provisional patent application no.
60/087,880, filed on June 3, 1998.

This application is related to co-pending U.S. patent application no. 09/224,682, filed
10   on December 31, 1998.

FIELD OF THE INVENTION

The present invention relates generally to securing cryptographic systems against
external attacks and, more specifically, to the minimization and masking of useful
15   information available by external monitoring of cryptographic operations.

BACKGROUND OF THE INVENTION

As described in U.S. Patent 4,908,038 to Matsumura et al., cryptographic devices can
be attacked using information gathered by observing the timing of comparison operations
20   performed by such devices during their operation. For example, if a MAC (Message
Authentication Code) algorithm is strong and the key is secure, forging a MAC should
require $O(2^n)$ attempts (where n is the MAC length in bits), but a device using a vulnerable
MAC validation process is vulnerable to an $O(n)$ timing attack.

If timing is the only source of leaked information, securing the device is often
25   relatively straightforward. Previously known countermeasures to attacks involving
information leaking from cryptosystems employ large and often expensive physical shielding
and/or careful filtering of inputs and outputs (e.g., U.S. government Tempest specifications).
Unfortunately, these techniques are difficult to apply in constrained engineering
environments. For example, physical constraints (such as size and weight), cost, and the need
30   to conserve power can often prevent the use of such techniques. It is also known to use
certain computational techniques (e.g., see Matsumura, above, or P. Kocher, "Timing Attacks
on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems," Advances in
Cryptology – CRYPTO '96, Springer-Verlag, 1996, pages 104-113) to equalize timing.

¯5   However, sources of information leakage other than timing (e.g., a device's power

consumption) provide other avenues of attack. Indeed, Matsumara's timing equalization

system itself can be vulnerable to non-timing attacks, for example by analyzing power

consumption to detect the start of processing delays. It would therefore be advantageous to

protect the devices' internal operations themselves instead of (or in addition to) simply

10   externally masking the devices' timing (or other) fluctuations.

        The present invention includes countermeasures that can be incorporated into software

and/or hardware, to provide improved protection at relatively low cost. Thus, the invention

could be used in place of (or in addition to) traditional countermeasures. For example, the

present invention can be implemented in smartcards and other highly constrained

15   environments where physical shielding and other protection measures cannot be readily

applied.

SUMMARY OF THE INVENTION

        The use of unpredictable information to minimize leakage from smartcards and other

20   cryptosystems is disclosed.

        According to one approach, the present invention provides techniques for modifying

the computational processes in implementations of cryptographic algorithms to incorporate

new random information, beyond the input parameters that are traditionally used, while still

producing desired results. Definitions and standards for cryptographic algorithms require that

25   implementations of such algorithms produce specific outputs from given inputs. For

example, implementations of the Data Encryption Standard (DES) defined in National Bureau

of Standards Federal Information Processing Standard Publication 46 (Jan. 1977) should

encrypt the message 0011223344556677 with the key 0123456789ABCDEF (with standard

odd DES key parity bits) to produce the ciphertext CADB6782EE2B4823. However,

30   implementers of this and other algorithms can choose the particular processing steps used to

transform the inputs into the outputs. Thus, by modifying the computational processes to

incorporate new random information, secret information that might be sought by an attacker

(such as the key or other secrets) can be concealed within or among random (or otherwise

unpredictable) information incorporated into the cryptographic operations. Information

35   leaked during the system's operation will then be correlated to the unpredictable state

5   information (or noise), making leaked information less useful to attackers. Said another way,

leaked information can be made effectively uncorrelated (or less correlated) to the device's

secrets. Some particular embodiments of this general approach will be described below. One

embodiment of the invention also provides for the added unpredictable information to be

updated frequently to prevent attackers from using monitoring attacks to determine the state

10   information itself.

An attacker's measurements of an operating device are often imperfect, and contain

both information that is useful ("signal") and information that hinders or is irrelevant to

interpretation of the signal ("noise"). (In addition, there may be irrelevant components of the

measurements, such as predictable information, that neither helps nor hinders attacks.) To

15   increase the difficulty of attack, one embodiment of the present invention increases the

amount of noise in attackers' measurements and/or increases the signal complexity.

Still other embodiments of the general technique include software- and hardware-

implementable clock skipping (to prevent the temporal correlation of specific operations with

clock transitions provided by or observable by attackers), symmetric permutation blinding,

20   and the introduction of entropy into the order of cryptographic operations. Such techniques

are usable to prevent attackers from correlating observations with specific events within the

cryptosystem's operation.

All of the foregoing will be explained in greater detail with respect to the figures and

detailed description of the invention, below.

25

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 illustrates an exemplary apparatus for introducing noise into a cryptosystem.

FIG. 2 illustrates an exemplary apparatus for implementing clock skipping.

30

DETAILED DESCRIPTION OF THE INVENTION

The following sections describe various embodiments of a general technique of using

unpredictable information to protect cryptographic systems (cryptosystems) against external

5   monitoring attacks. Although the embodiments differ in the details of their implementations, those skilled in the art will appreciate the fundamental commonality in their essential operation – using randomness or other sources of unpredictability to decorrelate secret information from externally monitorable signals in such a way that deters external monitoring attacks (including those involving statistical accumulation and analysis of collected data)

10   upon cryptographic systems.

Reduction of Signal-to-Noise Ratios

Unless noted otherwise, it shall be assumed herein that leakage (or the reducing,

15   masking, or minimizing thereof) refers to the leakage (or the reducing, masking, or minimizing thereof) of any information that is potentially useful to an attacker trying determine secret information. Thus, the leaked information includes the secret information itself, but also other information pertaining to that secret information. Of course, the attacked device may also leak information, such as information correlated to its internal processing

20   operations, that is not useful to attackers. However, such leakage of non-useful information is not relevant to this description of the present invention.

To obtain a secret key from a cryptosystem that leaks information, an attacker can gather data by observing a series of operations, perform statistical analysis on the observations, and use the results to determine the key. In a common situation, an attacker

25   monitors a physical property, such as power consumption, of a secure token as it performs a cryptographic operation. The attacker collects a small amount of data related to the key each time the token is observed performing a cryptographic operation involving the key. The attacker increases the amount of information known about the key by collecting and statistically correlating (or combining) data from multiple observations of the token as it

30   performs operations involving the key (or a related key).

In the case of a cryptosystem which is leaking information, such observations may contain signal (i.e., information correlated usefully to the key). However, such observations also contain noise (i.e., information and error that hinder or are irrelevant to determination of the key). The quality of the information gained from these observations is characterized by a

35   "signal to noise" (or S/N) ratio, which is a measure of the magnitude of the signal compared to the amount of noise.

The number of operations that the attacker must analyze to recover the key depends on the measurement and analysis techniques, but is generally inversely proportional to the square of the S/N ratio. The constant of proportionality also depends upon the amount of confidence the attacker requires. For example, a relatively low confidence level may be acceptable to an attacker willing to do an optimized brute force search using statistical information about key bit values. Decreasing the signal by a factor of 15 and increasing the amount of measurement noise by a factor of 20 will reduce the signal-to-noise ratio by a factor of 300. This will generally mean that an attacker will require roughly 90,000 times as many observations to extract the same amount of information about the key. An attack requiring 1,000 observations to recover a key before the S/N reduction would now require on the order of 90 million observations to gain the same level of confidence in the recovered key.

Thus, one approach according to the general technique of using unpredictable information to protect cryptosystems against external monitoring attacks is to implement cryptographic protocols so as to produce unpredictable state information, thereby increasing the number of observations required by an attacker to compromise a key. By reducing the available signal size and/or increasing the amount of error, noise, and uncertainty in attackers' measurements, a system designer can make the so-called work function (effort required) to break a system larger.

The system can be made even more secure by making the number of samples required to gain any significant amount of useful key information exceed the maximum number of transactions that can be performed using the key, exceed the number of transactions that can be performed by the device (e.g., before the key expires), or else be so large that monitoring attacks are comparable to (or of greater difficulty than) brute force and other known attacks. For example, consider a system programmed to self-destruct after one million operations -- well beyond the expected operational life of most smartcards. If a design not using the present invention requires five operations to break, and the present invention reduces the signal-to-noise ratio by a factor of 1000, the number of operations required to break the system (i.e., isolate the signal or key from the noise) might increase by a factor of roughly one million (i.e., to approximately 5 million) exceeding the lifetime of the secret or the device. Thus, attackers will be unable to collect enough measurements to compromise the secret.

5   **Random Noise Generation**

An exemplary apparatus for introducing noise into a cryptosystem is illustrated in

FIG. 1. In FIG. 1, noise production system 100 includes randomness source 101, noise

processing module 102 (such as, without limitation, a linear feedback shift register or a hash

10   function-based compression function), activation controller 103, digital/analog converter 104,

and noise production module (105). Other noise production systems including none, any, or

all of the components of FIG. 1 can also be used within the scope of the present invention.

Randomness source 101 creates the initial noise used to generate unpredictable

information. Randomness source 101 can be implemented in hardware or software. It is

15   preferable that the random number generator be implemented in hardware because hardware

implementations typically maintain less state information that can be subject to attack. If

random numbers are generated via software, care should be taken to ensure that attackers

cannot compromise the random number generator state and predict future random number

generator outputs. For example, to help make a software random number generator resist

20   external monitoring attacks, an implementation may incorporate extra state information and

update its state frequently. Of course, as will be appreciated by those skilled in the art, truly

random numbers are not always necessary or available. Therefore, as used herein, any term

described as "random" will be understood to include truly random, and also pseudorandom or

otherwise unpredictable, information suitable to, and depending on, the nature of the

25   particular application at hand.

Where randomness source 101 is an analog source, its output is first converted to

digital form, for example using digital/analog converter 104. The digital output produced by

randomness source 101 or digital/analog converter 104 is then provided as an input to noise

processing module 102. Noise processing module 102 converts the initial noise (which may

30   be biased or have other nonrandom characteristics) into either statistically random noise or

noise with desired characteristics (for example, random but with a nonlinear statistical

distribution).

Many cryptosystems spend a relatively small fraction of total processing time

performing security-critical operations. Therefore, the activation controller 103 can be

35   configured so that the noise production process is activated during operations in which

security is important (such as, without limitation, encryption, decryption, digital signing, data

~5   comparison, MAC verification, code verification, audit log updating, EEPROM update, and

key changing), but is deactivated during non-security critical operations. A noise production

activation control can thus greatly reduce many of the potential disadvantages of such a noise

system (such as increased power consumption, reduced performance, increased

electromagnetic radiation, decreased reliability, increased heat production, etc.). Activation

10   controller 103 can be implemented in any of a variety of ways, including without limitation in

a microprocessor cryptographic accelerator, or other well-known controller device that

disables power to one or more elements of noise production system 100, forces the output of

randomness source 101 (or mixer) to a particular value, forces the input or output of

digital/analog converter 104 to a particular value, or disables noise production module 105.

15   When activation controller 103 enables noise production system 100, random output

from noise processing module 102 is provided to digital/analog (D/A) converter 104. The

D/A output is provided to noise production module 105, which is configured to sink power,

produce electromagnetic radiation, or otherwise introduce noise into attackers' measurements,

where the noise produced is a function of the D/A input. The noise production module thus

20   introduces noise into attackers' measurements, increasing the difficulty of external

monitoring attacks. Digital/analog conversion methods are known in the background art, and

need not be described in detail here. For example, an array of current sources (e.g.,

transistors) and/or current sinks (e.g., resistors), as well as many other well known techniques

can be used.

25   In an embodiment where randomness source 101 is an analog noise source, noise

production module 105 can operate using the output of randomness source 101 as a direct

input. Activation controller 103 can then operate by regulating the output of randomness

source 101 or enabling and disabling noise production module 105.

      To prevent noise from being observably correlated to clock transitions or other

30   externally-measurable events, multiple noise production modules may be deployed and

driven simultaneously from the same or different random sources. Alternatively, the noise

processing module can be used to combine outputs from multiple noise sources and/or

provide inputs to multiple noise production modules. Also, because microprocessor current

usage profiles (and other externally measurable characteristics such as E/M radiation) are

35   instruction-dependent and carry significant detail within each clock period, it may be

advantageous to drive noise production modules faster than (or independently from) the clock

5   rate applied to cryptosystem microprocessor. For example, noise production modules may

include delay lines that temporally isolate their outputs from those of the others, or they may

be clocked independently, or they may be free-running.

All of the foregoing components may be implemented separately or in various

combinations, using analog or digital techniques as appropriate. Those skilled in the art will

10   also appreciate that various of the components can be implemented in hardware, or even

software, although hardware implementations will generally provide greater security. For

example, the noise source can be integrated within the cryptosystem microprocessor itself. In

single-chip environments (such as smartcards and secure microprocessors), the noise source

and noise control circuitry can be integrated into the same chip that contains the

15   microprocessor, secure memory, I/O interface, etc.

The signal-to-noise reduction techniques described herein may be implemented for

use in various environments, including without limitation key management and storage

systems, cryptographic accelerators (e.g., hardware DES implementations, multipliers, fast

modular exponentiators, hash functions, etc.), nonvolatile memory (e.g., EEPROM, flash,

20   etc.), data communication interfaces, buses, and (as will be evident to one of ordinary skill in

the art) other computational devices and methods used in cryptographic operations.

Clock Skipping

25   Another approach to the general technique of using unpredictable information to

protect cryptosystems against external monitoring attacks involves what will be referred to

herein as clock skipping (or clock decorrelation).

During statistical attacks using power consumption or electromagnetic radiation,

attackers typically compare measurements from several different operations against each

30   other. For example, an attacker might make a sequence of observations by sampling the

target device's power consumption at 200 MHz during a 5ms portion of each of 1,000

cryptographic operations done by the target device. For this exemplary attack, 1,000

observations each containing 1,000,000 data points are thus collected. The attacker would

then align these measurements so that the data points corresponding to a single point of

35   interest can be compared and analyzed across a large number of observations.

5   Therefore, security can be improved by preventing attackers from locating points of interest within collected data sets and from identifying corresponding regions between observations. Indeed, causing an attacker to include incorrectly-aligned data is one way to decrease the effective signal-to-noise ratio of the attacker's data (see previous section), since the noise increases significantly (due to the inclusion of uncorrelated samples) and the useful

10   signal decreases (due to the presence of fewer good samples).

       Without accurate temporal alignment, the temporal resolution of the attacker's observations decreases greatly, making it much more difficult for the attacker to identify a signal containing fine structure. For example, a "1" bit in a secret or private cryptographic key might statistically result in a power feature consisting of a $1\mu A$ increase above average

15   for 2 μs followed immediately by a decrease to $2\mu A$ below average for 1 μs, while a "0" key bit might result in a power feature consisting of a $1\mu A$ decrease below average for 2 μs followed by a $2\mu A$ increase above average for 1 μs. Differentiating such signals is easy with sub-microsecond resolution, but can be extremely difficult or impossible with only millisecond resolution unless an extraordinarily large number of samples is taken. Of course,

20   small temporal alignment variations may not be able to conceal signal characteristics that are of large amplitude or of long duration (e.g., comparable to or larger than the size of the alignment variations). In general, then, poor temporal alignment will reduce an attacker's ability to identify fine variations within operations and significantly increase the number of measurements required for a successful attack.

25   Many conventional systems, including commonly available smartcards, simply use external clocks for their cryptographic operations -- even though attackers can freely observe and manipulate the external clock. This greatly facilitates the ability of attackers to make the measurements necessary to attack the system. One embodiment of the present invention uses clock skipping (or clock decorrelation) to inhibit such attacks by reducing attackers' ability to

30   predict the system state. Clock skipping involves decorrelating cryptographic operations from the normal (external) clock cycles by creating a separate, internal clock signal that is used to control processor timing during cryptographic operations. While externally-measurable characteristics (particularly power consumption and electromagnetic radiation) can reveal when some internal clock cycles occur, clock skipping will make them much more

35   difficult for an attacker to accurately locate points of interest in measurements, particularly if

5   noise is introduced into the signal using the techniques of the present invention. This will be

described in more detail below with respect to an exemplary embodiment of the invention

illustrated in FIG. 2.

Referring now to FIG. 2, random number generator 200 (which can be, but need not

be, implemented in hardware) is used to determine which clock cycles (or clock state

10   transitions) are to be used by microprocessor core 225. Random number generator 200

produces a stream of random (or pseudorandom) digital output bits or analog noise as random

output 205. Clock skipping module 240 then combines (as will be described below) random

output 205 with clock signal 220 received from external smartcard interface 210. Of course,

clock signal 220 can also originate from another source (for example, if the invention is

15   implemented in environments other than smartcards). In embodiments where random number

generator 200 itself uses an external clock signal (e.g., where a random bit is output on each

clock state transition), random number generator 200 can, but need not, use clock signal 220.

Within clock skipping module 240, random output 205 is used to select cycles of

clock signal 220 to skip in order to produce clock signal 260. Alternatively, random output

20   205 can be used to select the closest corresponding cycles of clock signal 220 to be used as

clock signal 260, or random output 205 can even be used as clock signal 260 itself. Still

other approaches are possible, as will be appreciated by those skilled in the art; the basic

point being that clock signal 260 be (partially or wholly) decorrelated from external clock

signal 220 via random output 205.

25   If desired, clock skipping module 240 can optionally apply a filter to clock signal 260

to ensure desired characteristics. For example, to ensure a minimum clock rate (as opposed to

a statistical average), a transition of clock signal 260 may be forced after more than a

threshold number of cycles of clock signal 260 have been skipped, either recently or

consecutively (e.g., a transition of clock signal 260 can be forced if clock signal 260 has not

30   changed during more than three transitions of clock signal 220.)

Additionally, clock skipping module 240 can optionally monitor the clock rate (of

either clock signal 220 or 260) to prevent attackers from stopping the clock and analyzing the

device in a halted state or from operating the device too quickly. When module 240 detects

such a clock fault, it can reset microprocessor core 225, clear memory 290 (which can be

35   nonvolatile RAM, such as battery-backed CMOS, EEPROM, flash memory, a hard disk, or

other such storage used to store the key and/or other information), clear the state of

5   cryptographic accelerator 280, and log the fault in memory 290. Methods and apparatuses for

detecting such clock faults are well known in the background art and need not be described in

detail here.

In an alternative embodiment, clock skipping module 240 and microprocessor 225 are

combined, such that random output 205 can force microprocessor 225 to skip clock cycles.

10   For example, when microprocessor 225 is directed to skip a clock cycle (such as when three

output bits equal to zero are received in random output 205), the result of the current or next

instruction (or clock cycle) executed by the microprocessor is discarded and repeated.

In all of the foregoing, it should be noted that the fraction of skipped clock cycles

does not need to be very large; for example and without limitation, even skipping as few as

15   one clock cycle in 20 (on average) will introduce significant measurement drift.

One consideration introduced by clock skipping is the effect on other functions of the

system besides the cryptographic operations. In particular, clock skipping may sometimes

adversely affect operations requiring regular clock cycles. For example, in many smartcards,

one bit is sent or received on a serial I/O (input/output) line every 372 cycles of the external

20   clock. (Thus, a 3.579545 MHz external clock is compatible with a serial communication rate

of 9600 bits per second.) However, with clock decorrelation, microprocessor 225 will operate

at a different clock rate governed by signal 260. A mismatch between the data

communications clock rate and the microprocessor clock rate may result, causing I/O errors

to occur. Consequently, in devices implementing clock skipping, it is often advantageous for

25   the microprocessor to be controlled by external clock 220 during I/O operations.

This can be implemented via clock skipping activation signal 230, which is used to

select between external clock signal 220 and the (modified) internal clock that would

otherwise be produced by clock skipping module 140. As with the noise generator activation

signal of FIG. 1, clock skipping activation signal 220 can be produced by a microprocessor or

30   any other control device that is capable of knowing when to apply (or not apply) the clock

skipping. Selection of whether or not to clock skip at any particular time can be performed

by many well-known techniques that need not be described in detail here. For example, in

the exemplary embodiment of FIG. 2, microprocessor 225 is well suited for such a task

because it is necessarily aware of I/O operations associated with the receipt of data signals

35   270. In general, when I/O is performed or when other non-security-critical operations are in

progress, microprocessor core 225 can assert control signal 230 to cause clock skipping

5   module 240 to ignore random output 205 and provide external clock signal 220 directly as

clock signal 260. Control signal 230 and the noise production activation control signal

described previously can, but need not be the same signal.

In an alternative solution to the synchronization failure problem, two separate clocks

are used. A conventional external clock signal is used for I/O and other processing, where

10   clock skipping is not needed to protect secret information. However, an internal clock signal,

preferably but not necessarily generated in the device (for example, produced using a ring

oscillator, which is well known in the background art), is used for internal (e.g.,

cryptographic) processing. Thus, internal operations need not proceed at a speed related to or

derived from the external clock rate. The internal clock may be distorted or skipped, for

15   example, as described above. Alternatively, or in addition, where an analog process is used to

generate the internal clock, significant sources of randomness can also be incorporated to

adjust the frequency, drift, and jitter of the clock signal to prevent accurate prediction of clock

state transitions. Clock signal selection can be performed by microprocessor 225 as

mentioned previously. Another technique, which is especially suitable for, but not limited to

20   smartcards, uses a UART (universal asynchronous receiver/transmitter) or other buffer

between the internally clocked region and the external I/O interface to ensure that

communications over the external serial I/O interface are clocked at a rate corresponding to

the externally-supplied clock but may be accessed reliably by internally-clocked circuits.

In yet another approach, the internally-generated clock signal can be derived from the

25   external clock signal. This can be performed via an analog phase-locked loop, which is well

known in the background art and need not be described in detail here. Such an internal clock

will be rather closely correlated with the external clock, and therefore not as effective against

attacks as the randomized clock signal or two separate clocks described previously. (Of

course, its effectiveness can be improved by optionally using clock skipping or analog (or

30   other) noise sources to adjust the frequency, drift, and jitter of its signal.) Also, when

synthesizing the internal clock from the external clock, the clock-derivation circuitry can be

configured to restrict the rate of the internal clock frequency, for example, to enforce a

minimum internal clock frequency so that attackers cannot stop the clock and attack the

device in the stopped state. The derived internal clock signal exhibits a number of useful

35   properties that will be described in the following paragraph.

5   One useful property of such a slightly decorrelated internal clock is that it may be

sufficiently close to the external clock that it may be used to control I/O rates reliably. In

addition, because a phase-locked loop can continue to produce a valid clock signal even if the

external clock changes or is removed, microprocessor 225 can continue operating so that it

can detect and even respond to attacks that involve halting, removing, or altering the external

10   clock while power is connected. The use of an internally-generated clock additionally

provides protection against attacks involving the introduction of errors into computations.

For example, jitter or high frequencies supplied to the external clock would cause harmless

communication errors, but would be prevented from causing erroneous computations.

Because a phase locked loop can produce an internal clock signal that is a multiple of the

15   external clock signal, it is possible to clock cryptographic operations at a rate faster than the

external clock, providing improved performance. In smartcards with challenging

performance requirements (such as those that run interpreted codes such as Java), this is an

added benefit.

All of the foregoing paragraphs describe various ways to generate a second, internal

20   clock signal: via randomization, via a separate clock, or via derivation from the external

clock. In all of these cases, the internal clock can also be used to monitor the external clock

to detect abnormalities introduced by attackers. Regardless of whether the clock is produced

internally or derived from the external clock, the microprocessor can have the option of

selecting between multiple clock modes. For example, a slower mode might be used if it has

25   a lower probability of computation error, a faster mode might be used when improved

performance is needed, and clock skipping or other clock distortion might be activated when

resistance to external monitoring attacks is desired.

Much of the foregoing has been described with respect to hardware techniques for

clock decorrelation (e.g., second clocks or phase-locked loops), but clock decorrelation can

30   also be effected by software as will be described below. This is useful, for example, where

the environment does not allow for hardware-based clock skipping. Alternatively, hardware

clock decorrelation could be supplemented with software-based clock decorrelation for even

greater protection in security critical code segments.

One efficient software-implementable technique for clock decorrelation takes

35   advantage of the fact that the amount of time used for a process with a conditional jump can

vary depending on whether or not the jump is actually performed. In such cases, inserting

branch delays can be used as a form of clock decorrelation. For example, the assembly language clock randomizer below uses a random generator to introduce clock variations that can help prevent accurate alignment by an attacker:

```
Assembly Language Clock Randomizer:

  [...]
  inp reg5,RANDOM_GENERATOR        # get a random byte
  add reg5,reg5                    # shift reg5 left once
  brc delay1                       # branch if carry
  nop                              # extra delay if bit is 0
delay1:                            # continue execution
  [...]                            # ...more code...
  add reg5,reg5                    # shift reg5 left again
  brc delay_2                      # branch if carry
    # put any code here            # obfuscating code/delay
delay2:                            # continue execution
  [...]                            # ...more code...
```

In an alternative embodiment, instead of using random information to determine whether to take a delay branch, the random information may be used to select between parallel code processes, such that the same cryptographic result will be produced regardless of which code process is selected but where the parallel processes perform different operations toward producing the result.

This section has described temporal obfuscation techniques that are useful in preventing reliable identification and alignment of specific features in measurements of cryptographic device characteristics such as power consumption and electromagnetic radiation. However, such techniques may not always be sufficient for preventing attacks based on timing, since introduced timing delays will have a predictable statistical distribution for which attackers can compensate. Another embodiment of the general technique of implementing cryptographic protocols using unpredictable information, described below, is useful in (but is not limited to) such contexts.

Execution Path and Operation Order Entropy

Another approach to the general technique of using unpredictable information to protect cryptosystems against external monitoring attacks involves the introduction of entropy into the order of processing operations or into the execution path while maintaining desired

5   functionality (such as compatibility with standard cryptographic algorithm definitions). More

specifically, a device can use a random number generator to cause unpredictability in the

order of performing a sequence of suboperations. If attackers cannot accurately determine the

order in which operations were performed, cross-correlation between samples becomes more

difficult or impossible. Consequently the data collected by an attacker effectively has a

10   significantly lower signal-to-noise ratio.

As an illustrative example of operation order entropy, consider a bit permutation.

Permutations are widely used in cryptography, for example in the Data Encryption Standard

and other cryptographic algorithms. The following C language pseudocode illustrates a

traditional method of implementing a permutation.

15

```
Input-Ordered Permutation (Background Art):

    void perm2(bool dataIn[64], bool dataOut[64], int table1[64]) {
        int i;

        for (i = 0; i < 64; i++) {
            dataOut[table1[i]] = dataIn[i];
        }
    }
```

This example is input-ordered, meaning that processing steps are performed in the

order (or inverse order) in which the input bits are supplied. In the example, input bit 0 is

permuted first, and input bit 63 is permuted last. Output-ordered permutations are also

20   commonly used in the background art. Provided that `table1` is a permutation (i.e., where

one element equals each of the values 0...63), the pseudocode below can be made output-

ordered by changing the statement inside the loop to read: "`dataOut[i] =`

`dataIn[table2[i]];`", where `table2` is output-ordered (i.e., `table2` is the inverse of

`table1` above such that `table1[table2[i]] = i`).

25   However, both output-ordered and input-ordered permutations can leak information

about the data they process. For example, in the input-ordered permutation, attackers'

measurements of loop iteration `i` will be correlated to `dataIn[i]`. In the output-ordered

permutation, the attackers' measurements of loop iteration `i` will be correlated to

`dataOut[i]`. An improved permutation method would thus be advantageous. One

30   exemplary implementation of such a method is shown in the table below. This high-entropy

5   permutation combines several previously-described aspects of the present invention, including without limitation order randomization (thus being neither input-ordered nor output-ordered) and blinding techniques (to conceal further the data being permuted).

```
Blinded High-Entropy Permutation:

  #define SWAP(a,b) { register int t = a; a = b; b = t; }
  #define LOOPCOUNT 128

  void perm3(bool dataIn[64], bool dataOut[64], int table[64]) {
      unsigned char trueRandom(void);        /* gives random byte */
      int i,p;
      int perm[64];
      bool b, temp[64];

      /* Initialize random permutation */
      for (i = 0; i < 64; i++) {
        perm[i] = i;
        temp[i] = trueRandom() & 1;
        dataOut[i] = trueRandom() & 1;
      }
      for (i = 0; i < LOOPCOUNT; i++) {
        p = trueRandom() & 63;              /* random number mod 64 */
        SWAP(perm[p], perm[i&63]);
      }

      /* Blind: temp=blinded input, dataOut=unblinding factor */
      for (i = 0; i < 64; i++) {
        p = perm[i];
        b = (bool)(trueRandom() & 1);
        temp[p] = dataIn[p] ^ b;
        dataOut[table[p]] = b;
      }
      for (i = 0; i < LOOPCOUNT; i++) {
        p = trueRandom() & 63;              /* random number mod 64 */
        SWAP(perm[p], perm[i&63]);
      }

      /* Perform the permutation on temp & unblind */
      for (i = 0; i < 64; i++) {
          p = perm[i];
          dataOut[table[p]] ^= temp[p];
          temp[p] = 0;
      }
  }
```

10   The magnitude of signals leaked due to variations in data values (e.g., registers and memory contents) is usually smaller (often by a factor of several orders of magnitude) than signals leaked due to branches and variations in the execution path. Therefore, the high-entropy permutation operation, above, uses a constant execution path to inhibit leakage via variations in the execution path.

5   The exemplary blinded randomized-order permutation operation includes four steps, which can be performed separately or simultaneously: initialization, blinding, permutation, and unblinding. Implementations using partial blinding, which operate on already-blinded values, or those with reduced security requirements will not require all steps.

Initialization of the blinded randomized-order permutation operation involves
10   constructing and randomizing a permutation table ("perm") for determining the bit order for operations. (Bit order permutation table "perm" randomizes the time at which any particular data bit is manipulated.) The bit order table is created in two passes, where the first assures that the table has the correct form (i.e., contains the numbers zero through 63), and the second introduces random order into the table. Because the process of constructing the bit order
15   table does not involve any secret inputs, the only security requirement for the process is that the final result be unknown to attackers. As illustrated, the first permutation table initialization loop can also place random values into dataOut and temp to help whiten any leaked signals when data values are first stored in these arrays. Finally, although it is not required, more than 64 iterations of the randomization loop are used to ensure that any
20   statistical biases remaining after the randomization loop are insignificantly small.

The next section of the code performs the blinding operation. First, for each loop iteration, a random number generator produces a random blinding bit. The temporary buffer (temp) is initialized with the XOR of the random bit and an input data bit, where the input data bit is selected according to the table (perm) constructed previously. Additionally, the
25   output buffer (dataOut) is initialized with the blinding bit, where the dataOut bit is the result of using the input permutation table to operate on the index to temp. The second part of the blinding process re-randomizes the bit order permutation table (perm).

The last section performs the final bit permutation and unblinding steps. Input bits are loaded in the order specified by the table (perm), permuted according to the (non-secret)
30   externally-specified permutation table (table), and XORed onto the destination table (dataOut).

Note that the leak-minimized permutation operation described dramatically reduces the amount of information leaked from a permutation operation, but is not necessarily expected to reduce such leakage to zero. The input data to the function arrives in fixed order
35   and unblinded form, and the output is similarly supplied unblinded in fixed order. Consequently, two or more measurements from the same transaction might (for example) be

5   correlated to each other such that the strength or sign of the correlation is a function of one or more input or output data bits. If inputs and/or outputs must be kept secret or if multiple permutations are to be performed on the same secret data (for example, through a multi-step operation such as encryption), operands can be maintained in a blinded state during processing, to be (partially or completely) reconstituted only when nonlinear operations must

10   be performed or at the end of the computation.

Note that many variations on the process described are possible, as will be understood to those skilled in the art. For example and without limitation, the number of bits manipulated does not need to equal 64, the order of steps may be changed, steps can be removed for simplified implementations (such as those that are not subject to some attacks),

15   steps can be modified, different permutation generation and update processes can be used, and additional steps can be added.

Other Considerations

20   Cryptographic operations should normally be checked to ensure that incorrect computations do not compromise keys or enable other attacks. Cryptographic implementations of the present invention can be, and in a preferred embodiment are, combined with error-detection and/or error-correction logic to ensure that cryptographic operations are performed correctly. For example, a simple and effective technique is to

25   perform cryptographic operations twice, ideally using two independent hardware processors and/or software implementations, with a comparison operation performed at the end to verify that both produce identical results. If the results produced by the two units do not match, the failed comparison will prevent the defective processing result from being used. In situations where security is more important than reliability, if the compare operation ever fails (or fails

30   too many times) the device may self-destruct (such as by deleting internal keys) or disable itself. For example, a device might erase its key storage memory if either two defective DES operations occur sequentially or five defective DES results occur during the lifetime of the device. In some cryptosystems, full redundancy is not necessary. For example, with RSA, methods are known in the background art for self-checking functions that can be incorporated

35   into the cryptosystem implementation (e.g., RSA signatures can be verified after digital signing operations).

5   Detection of conditions likely to cause incorrect results may also be used. In particular, active or passive sensors to detect unusually high or low voltages, high-frequency noise on voltage or signal inputs, exposure to electromagnetic fields and radiation, and physical tampering may be employed. Inappropriate operating conditions can (for example) trigger the device to reset, delete secrets, or self-destruct.

10   Self-diagnostic functions such as a POST (power-on-self-test) should also be incorporated to verify that cryptographic functions have not been damaged. In cases where an ATR (answer-to-reset) must be provided before a comprehensive self-test can be completed, the self-test can be deferred until after completion of the first transaction or until a sufficient idle period is encountered. For example, a flag indicating successful POST

15   completion can be cleared upon initialization. While the card is waiting for a command from the host system, it can attempt the POST. Any I/O received during the POST will cause an interrupt, which will cancel the POST (leaving the POST-completed flag at zero). If any cryptographic function is called, the device will check the POST flag and (if it is not set) perform the POST before doing any cryptographic operations.

20

Conclusions

          The present invention is extremely useful for improving security, particularly in environments and applications with difficult engineering requirements, by enabling the

25   construction of devices that are significantly more resistant to attack than devices of similar cost and complexity that do not use the present invention. Also, multiple security techniques may be required to make a system secure. For example, leak minimization and obfuscation may be used in conjunction with other security methods or countermeasures.

          As those skilled in the art will appreciate, the techniques described above are not

30   limited to particular host environments or form factors. Rather, they may be used in a wide variety of applications, including without limitation: cryptographic smartcards of all kinds including without limitation smartcards substantially compliant with ISO 7816-1, ISO 7816-2, and ISO 7816-3 ("ISO 7816-compliant smartcards"); contactless and proximity-based smartcards and cryptographic tokens; stored value cards and systems; cryptographically

35   secured credit and debit cards; customer loyalty cards and systems; cryptographically authenticated credit cards; cryptographic accelerators; gambling and wagering systems;

5 secure cryptographic chips; tamper-resistant microprocessors; software programs (including without limitation programs for use on personal computers, servers, etc. and programs that can be loaded onto or embedded within cryptographic devices); key management devices; banking key management systems; secure web servers; electronic payment systems; micropayment systems and meters; prepaid telephone cards; cryptographic identification

10 cards and other identity verification systems; systems for electronic funds transfer; automatic teller machines; point of sale terminals; certificate issuance systems; electronic badges; door entry systems; physical locks of all kinds using cryptographic keys; systems for decrypting television signals (including without limitation, broadcast television, satellite television, and cable television); systems for decrypting enciphered music and other audio content (including

15 music distributed over computer networks); systems for protecting video signals of all kinds; intellectual property protection and copy protection systems (such as those used to prevent unauthorized copying or use of movies, audio content, computer programs, video games, images, text, databases, etc.); cellular telephone scrambling and authentication systems (including telephone authentication smartcards); secure telephones (including key storage

20 devices for such telephones); cryptographic PCMCIA cards; portable cryptographic tokens; and cryptographic data auditing systems. All of the foregoing illustrates exemplary embodiments and applications of the invention, from which related variations, enhancements and modifications will be apparent without departing from the spirit and scope of the invention. Therefore, the invention should not be limited to the foregoing disclosure, but

25 rather construed by the claims appended hereto.

CLAIMS

What is claimed is:

1. 1.   A cryptographic processing device for securely performing a cryptographic processing
2. operation in a manner resistant to discovery of a secret by external monitoring,
3. comprising:
4.   (a)   an input interface for receiving a quantity to be cryptographically processed;
5.   (b)   a source of unpredictable information;
6.   (c)   a processor:
7.   (i)   connected to said input interface for receiving and cryptographically
8.   processing said quantity,
9.   (ii)   configured to use said unpredictable information to conceal a
10.   correlation between externally monitorable signals and said secret
11.   during said processing of said quantity; and
12.   (d)   an output interface for outputting said cryptographically processed quantity to
13.   a recipient thereof.

1. 2.   The device of Claim 1 wherein said input interface and said output interface are the
2. same element.

1. 3.   The device of Claim 1 wherein said processor is configured to use said unpredictable
2. information in a manner such that said cryptographically processed quantity is
3. independent of said unpredictable information.

1. 4.   The device of Claim 3 wherein said cryptographic processing operation includes a
2. sequence of instructions and wherein said unpredictable information is used to modify
3. said sequence.

1. 5.   The device of claim 4 wherein said cryptographic processing operation includes
2. transforming a message with the Data Encryption Standard (DES).

1   6.   The device of Claim 3 wherein said cryptographic processing operation includes a
2   permutation and said unpredictable information is used to randomizing the order of
3   said permutation.

1   7.   The device of claim 3 wherein:
2   (a)   said device is implemented on a single microchip;
3   (b)   said concealed correlation is a correlation between said microchip's power
4   consumption and said processing; and
5   (c)   said correlation is concealed by expending additional electricity in said
6   microchip during said processing.

1   8.   The device of claim 7 including program logic to activate said expending during said
2   processing.

1   9.   The device of claim 8 including
2   (a)   program logic implementing said source of unpredictable information; and
3   (b)   program logic to transmit said unpredictable information to an additional
4   power expending circuit contained in said microchip.

1   10. The device of claim 1 further including:
2   (a)   a hardware-implemented noise production subunit connected to said source of
3   unpredictable information and configured to expend unpredictable amounts of
4   electricity based on the output of said source of unpredictable information; and
5   (b)   a activation controller, which may be activated by software contained in said
6   device, to activate and deactivate said expending of unpredictable amounts of
7   electricity.

1   11. The device of claim 10 wherein said source of unpredictable information is a
2   hardware-implemented random number generator, and wherein said noise production
3   subunit includes a digital-to-analog converter.

1   12.   A cryptographic processing device for securely performing a cryptographic processing

2   operation in a manner resistant to discovery of a secret by external measurement of

3   said device's power consumption, comprising:

4   (a)   an input interface for receiving a quantity to be cryptographically processed;

5   (b)   an input interface for receiving a variable amount of power, said power

6   consumption varying measurably during said performance of said operation;

7   (c)   a processor connected to said input interface for receiving and

8   cryptographically processing said quantity; and

9   (d)   a noise production system for introducing noise into said measurement of said

10   power consumption.

1   13.   The device of Claim 12 wherein said noise production system comprises:

2   (a)   a source of randomness for generating initial noise having a random

3   characteristic;

4   (b)   a noise processing module for improving the random characteristic of said

5   initial noise; and

6   (c)   a noise production module configured to vary said power consumption based

7   on an output of said noise processing module.

1   14.   The device of Claim 13 wherein said noise production system is connected to said

2   processor and is selectively operable under the control of said processor.

1   15.   A cryptographic processing device for securely performing a cryptographic processing

2   operation in a manner resistant to discovery of a secret by external monitoring of said

3   device's power consumption, comprising:

4   (a)   an input/output interface for receiving data to be cryptographically processed;

5   (b)   an oscillator generating a first clock signal;

6   (c)   an input interface for receiving a variable amount of power, said power

7   consumption varying measurably during said performance of said operation;

8   (d)   a source of unpredictable information;

9   (e)   a clock decorrelator coupled to said source of unpredictable information for

10   generating a second clock signal from said first clock signal using said

```
r1          unpredictable information, such that said second clock signal cannot be

12          reliably predicted from said first clock signal; and

13   (f)    a processor:

14          (i)    clocked by said second clock signal,

15          (ii)   configured to cryptographically processing said data, and

16          (iii)  configured to output said cryptographically processed data using said

17                 input/output interface.
```

```
1    16.  A cryptographic processing device for securely performing a cryptographic processing

2         operation in a manner resistant to discovery of a secret by external monitoring of said

3         device's power consumption, comprising:

4    (a)  an input/output interface for receiving data to be cryptographically processed;

5    (b)  an input interface for receiving an external clock signal;

6    (c)  an input interface for receiving a variable amount of power, said power

7         consumption varying measurably during said performance of said operation;

8    (d)  a source of unpredictable information;

9    (e)  a clock decorrelator coupled to said source of unpredictable information for

10        generating an internal clock signal from said external clock signal using said

11        unpredictable information, such that said internal clock signal cannot be

12        reliably predicted from said external clock signal; and

13   (f)  a processor:

14        (i)    clocked by said internal clock signal,

15        (ii)   configured to cryptographically processing said data, and

16        (iii)  configured to output said cryptographically processed data using said

17               input/output interface.
```

```
1    17.  The device of Claim 16 wherein said clock decorrelator comprises a clock skipping

2         module which selects a subset of the cycles of said external clock signal to use as said

3         internal clock signal based on said unpredictable information.
```

```
1    18.  The device of Claim 16 wherein said source of unpredictable information comprises a

2         hardware random number generator.
```

1 19. The device of Claim 16 further comprising a monitor for detecting a clock fault in
2   said external clock signal and preventing said processor from processing said quantity
3   if said clock fault is detected.

1 20. The device of Claim 16 wherein said clock decorrelator is selectively operable under
2   the control of said processor.

1 21. The device of Claim 16 wherein said clock decorrelator is selectively operable such
2   that said clock decorrelator is disabled when data is being transferred across said
3   input/output interface and enabled during said cryptographic processing operation.

1 22. The device of Claim 16 further comprising a noise production system connected to
2   said processor for introducing noise into said measurement of the power consumption
3   by consuming a random amount of power during said cryptographic processing
4   operation.

1 23. A device according to Claims 1, 5, 8, 10, 12, 15, 16, or 21 wherein said device
2   comprises an ISO 7816 compliant smartcard.

1 24. A method of securely performing a cryptographic processing operation in a manner
2   resistant to discovery of a secret within a cryptographic processing device by external
3   monitoring, comprising:
4   (a)   receiving a quantity to be cryptographically processed;
5   (b)   generating unpredictable information;
6   (c)   using said unpredictable information while processing said quantity to conceal
7   a correlation between externally monitorable signals and said secret; and
8   (d)   outputting said cryptographically processed quantity to a recipient thereof.

1 25. The method of claim 24, wherein said step (c) includes using said unpredictable
2   information to select between:

3   (c)(1) performing a computation and incorporating the result of said computation in

4   said cryptographic processing; and

5   (c)(2) performing a computation whose output is not incorporated in said

6   cryptographic processing.

1 26.   The method of claim 25 where said selecting is performed in software.

1 27.   The method of claim 25 where said selecting is performed in hardware on an

2   integrated circuit including a microprocessor.

1 28.   The method of Claim 24 where said unpredictable information is used to select a code

2   process from a plurality of code processes, where said selected code process is

3   involved in said cryptographic processing, but where the value of said outputted

4   quantity is independent of which of said code processes was selected.

1 29.   The method of Claim 24 wherein said step of using said unpredictable information

2   comprises a step of using said unpredictable information in a manner such that said

3   cryptographically processed quantity is independent of said unpredictable

4   information.

1 30.   The method of Claim 29 wherein said cryptographic processing operation includes a

2   sequence of instructions and wherein said step of using said unpredictable

3   information comprises a step of using said unpredictable information to modify said

4   sequence.

1 31.   The method of Claim 29 wherein said cryptographic processing operation includes a

2   permutation and wherein said step of using said unpredictable information comprises

3   a step of using said unpredictable information to randomize the order of said

4   permutation.

32. A method of securely performing a cryptographic processing operation in a manner resistant to discovery of a secret within a cryptographic processing device by external monitoring of said device's power consumption, comprising:

    (a) receiving a variable amount of power, said power consumption varying measurably during said performance of said operation;

    (b) receiving a quantity to be cryptographically processed;

    (c) introducing noise into said measurement of said power consumption while processing said quantity; and

    (d) outputting said cryptographically processed quantity to a recipient thereof.

33. The method of Claim 32 wherein said step of introducing noise comprises:

    (a) generating initial noise having a random characteristic;

    (b) improving the random characteristic of said initial noise; and

    (c) varying said power consumption based on said improved initial noise.

34. A method of securely performing a cryptographic processing operation in a manner resistant to discovery of a secret within a cryptographic processing device by external monitoring of said device's power consumption, comprising:

    (a) receiving a variable amount of power, said power consumption varying measurably during said performance of said operation;

    (b) generating a first clock signal;

    (c) receiving data to be cryptographically processed;

    (d) generating unpredictable information;

    (e) generating a second clock signal from said first clock signal using said unpredictable information, such that said second clock signal cannot be reliably predicted from said first clock signal;

    (f) processing said data using said second clock signal; and

    (g) outputting said cryptographically processed quantity to a recipient thereof.

35. A method of securely performing a cryptographic processing operation in a manner resistant to discovery of a secret within a cryptographic processing device by external monitoring of said device's power consumption, comprising:

4   (a)   receiving a variable amount of power, said power consumption varying

5   measurably during said performance of said operation;

6   (b)   receiving an external clock signal;

7   (c)   receiving data to be cryptographically processed;

8   (d)   generating unpredictable information;

9   (e)   generating an internal clock signal from said external clock signal using said

10   unpredictable information, such that said external clock signal cannot be

11   reliably predicted from said internal clock signal;

12   (f)   processing said data using said internal clock signal; and

13   (g)   outputting said cryptographically processed quantity to a recipient thereof.

1   36.   The method of Claim 35 wherein said step of generating said internal clock signal

2   comprises a step of selecting a subset of the cycles of said external clock signal to use

3   as said internal clock signal based on said unpredictable information.

1   37.   The method of Claim 35 wherein said step of generating unpredictable information

2   comprises a step of generating a random number.

1   38.   The method of Claim 35 further comprising a step of monitoring for a clock fault in

2   said external clock signal and a step of preventing said processor from outputting said

3   cryptographically processed quantity if said clock fault is detected.

1   39.   The method of Claim 35 further comprising a step of introducing noise into said

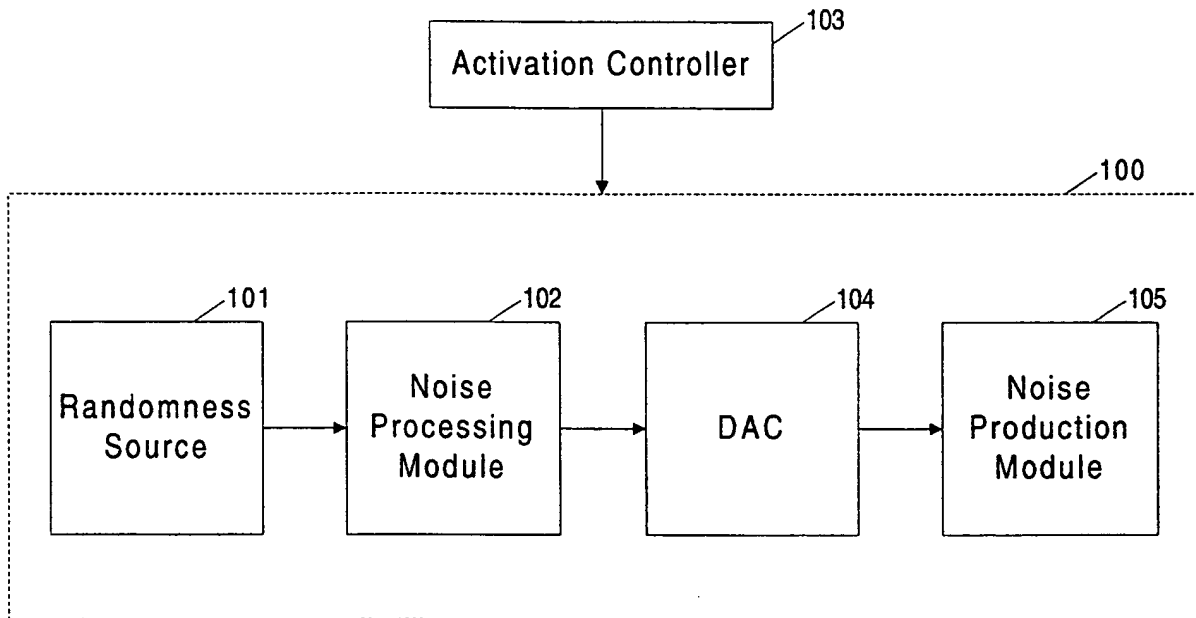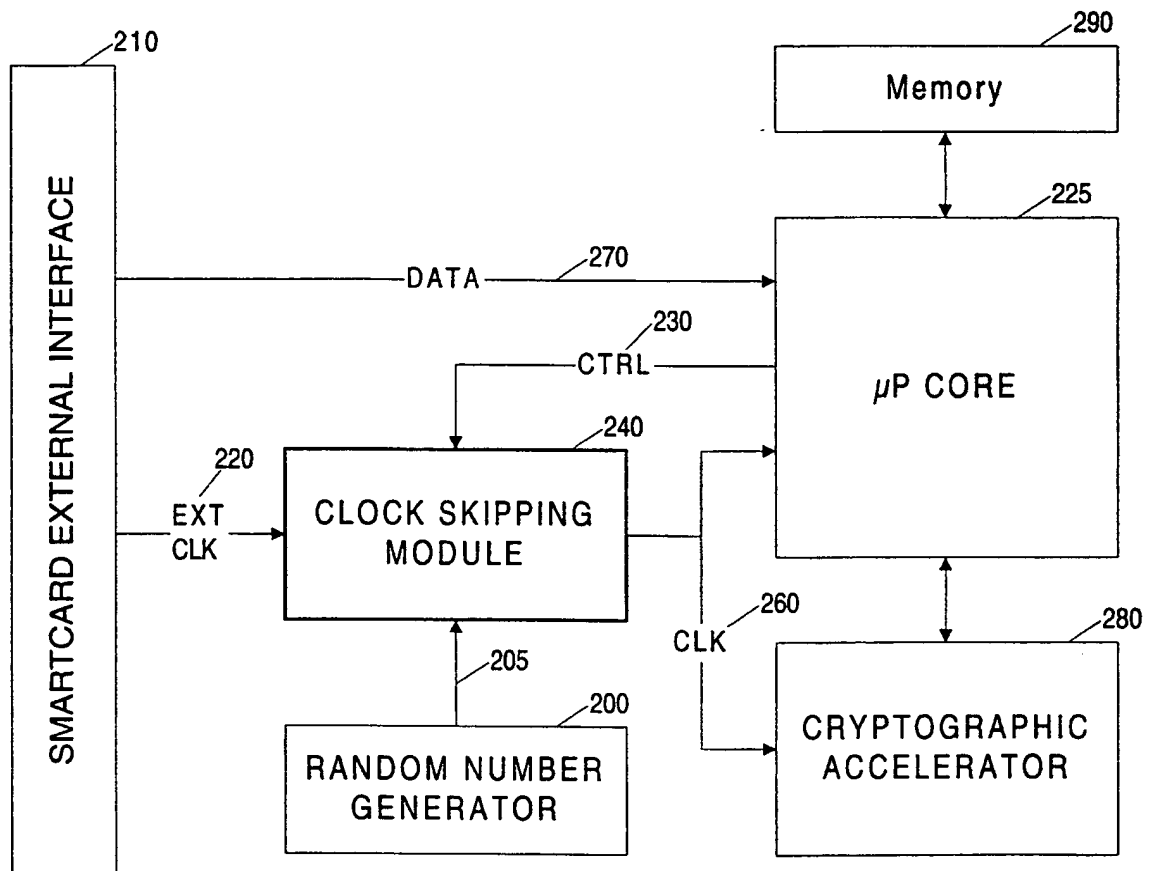2   measurement of the power consumption.

FIG. 1

Activation Controller — 103

— 100

Randomness Source — 101

Noise Processing Module — 102

DAC — 104

Noise Production Module — 105

FIG. 2

INTERNATIONAL SEARCH REPORT

A. CLASSIFICATION OF SUBJECT MATTER

IPC(6) :H04K 1/00
US CL :380/1

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 380/1,4,28,29,48; 364/717

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

APS (timing attacks, external attacks, information leakage, random noise generation)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 5,404,402 A (SPRUNK) 04 April 1995, see background and summary	1-5, 24
--		----------------
Y		6-23,25-39
X	US 5,539,827 A (LIU) 23 July 1996, see abstract and col. 4, lines 46-56.	1-4
Y	US 4,905,176 A (SCHULZ) 27 February 1990, see summary	6-23,25-39

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

*	Special categories of cited documents:	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be of particular relevance		
"E"	earlier document published on or after the international filing date	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"O"	document referring to an oral disclosure, use, exhibition or other means		
"P"	document published prior to the international filing date but later than the priority date claimed	"&"	document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
26 AUGUST 1999	10 SEP 1999

Name and mailing address of the ISA/US	Authorized officer
Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231.	Gail Hayes *James R. Matthew*
Facsimile No. (703) 305-3230	Telephone No. (703) 306-5539

Form PCT/ISA/210 (second sheet)(July 1992)★

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 4,905,176 A (SCHULZ) 27 February 1990, see summary	6-23,25-39